The Problem of Compatible Representatives

by Donald E. Knuth* and Arvind Raghunathan**

Abstract. The purpose of this note is to attach a name to a natural class of combinatorial problems and to point out that this class includes many important special cases. We also show that a simple problem of placing nonoverlapping labels on a rectangular map is NP-complete.

Keywords backtracking, coloring, compatibility, independent sets, mapmaking, matching, NP-complete, preclusion, radio communication

AMS (MOS) subject classifications 68R99 (Discrete mathematics in relation to computer science); 90C27 (Combinatorial programming)

Abbreviated title Compatible Representatives

---

* Computer Science Department, Stanford University; research supported in part by National Science Foundation grant CCR–8610181.

** University of California, Davis, Division of Computer Science; research supported in part by Semiconductor Research Corporation grant SRC–82–11–008.

The Problem of Compatible Representatives
by Donald E. Knuth and Arvind Raghunathan

Many combinatorial tasks can be formulated in the following way: Is there a sequence $(x_1, x_2, \ldots, x_n)$ such that $x_j \in A_j$ for all j, and x_j is compatible with x_k for all $j < k$? Here $A_1, A_2, \ldots, A_n$ are given sets, and "compatibility" is a given relation on $A_1 \cup A_2 \cup \ldots \cup A_n$.

This problem is NP-hard in general. For example, if all sets A_j are the same, and if compatibility is a symmetric, irreflexive relation, a sequence of compatible representatives is nothing but an n-clique in the compatibility graph.

The problem of coloring a graph G with c colors is another NP-hard special case of the general compatibility question. Let A_j be the the set of pairs $\{(j,1), \ldots, (j,c)\}$, and say that (j,a) is compatible with (k,b) if either $a \neq b$ or v_j is not adjacent to v_k in G, where the vertices of G are $\{v_1, \ldots, v_n\}$. Then a sequence of compatible representatives is essentially a c-coloring of G. Therefore the problem is NP-hard for all $c \geq 3$.

On the other hand, the compatibility problem also has important special cases that are efficiently solvable. If the compatibility relation is '$\neq$', then a solution sequence $(x_1, \ldots, x_n)$ is traditionally called a system of distinct representatives [4] [3, Chapter 5], and the problem of finding such systems is well known to be equivalent to bipartite matching. Indeed, if the compatibility relation is the complement of any equivalence relation, a sequence $(x_1, x_2, \ldots, x_n)$ of compatible representatives exists if and only if there is a matching of cardinality n in a bipartite graph on the vertices $\{v_1, \ldots, v_n, c_1, \ldots, c_m\}$, where $\{c_1, \ldots, c_m\}$ are the equivalence classes and we have v_j — c_k if and only if A_j contains an element of class c_k.

Another nice special case is equivalent to identifying increasing subsequences of a permutation. Let $\pi_1 \ldots \pi_m$ be a permutation of $\{1, \ldots, m\}$, and let A_j be the set of pairs $\{(j,1), \ldots, (j,m)\}$. Say that (j,a) is compatible with (k,b) if and only if $j < k$ and $\pi_a < \pi_b$. Then a compatible sequence $\bigl((1,a_1), \ldots, (n,a_n)\bigr)$ is equivalent to an increasing subsequence $(\pi_{a_1}, \ldots, \pi_{a_n})$ of $\pi_1 \ldots \pi_m$.

The example in the prevous paragraph illustrates that compatibility need not be a symmetric relation. But when the sets A_j are pairwise disjoint, as in that case, we could just as well assume that compatibility is symmetric and reflexive, since our definition of compatible representatives makes it immaterial whether elements x_j of A_j and x_k of A_k are compatible unless we have $j < k$.

There are, however, important special cases in which compatibility is asymmetric. Consider, for example, a scheduling problem in which A_j is a set of tasks that can be done at time j, and where x_j is compatible with x_k only when task x_j does not require the prior completion of x_k.

Cartographers face an interesting case of the general compatibility problem when they attach alphabetic labels to dots on a map. Let A_j represent the possible ways to place the name of city j, and let x_j be compatible with x_k when positions x_j and x_k do not overlap each other or otherwise mislead a potential reader. Then a good map should be a solution to the problem of compatible representatives.

Notice that the cartographic problem makes sense even if the sets A_j are infinite. The

task of placing disjoint labels is a fairly natural question of combinatorial geometry that does not appear to be a special case of any other well known problem.

In light of this discussion, it seems worthwhile to add the problem of compatible representatives to the class of "combinatorial problems that deserve a name," and to investigate heuristics and additional special cases that turn out to have efficient solutions.

Simple special cases. We have noted that the compatibility problem is equivalent to bipartite matching when incompatibility is an equivalence relation. The problem also has a polynomial-time solution when compatibility is transitive. Let $B_1 = A_1$, and for $j > 1$ let

$$B_j = \{\, y \in A_j \mid \exists x \in B_{j-1}\ (x \text{ compatible with } y) \,\}\,.$$

Then the compatibility problem has a solution if and only if B_n is nonempty. We can decide this in at most $\sum_{j=2}^{n} \|A_{j-1}\|\,\|A_j\|$ steps.

Another noteworthy special case occurs when each set A_j contains at most two elements. Then the compatibility problem is equivalent to an instance of `2SAT`: We can assume that $A_j = \{v_j, \overline{v}_j\}$; the clauses are $(\overline{\sigma}_j \vee \overline{\sigma}_k)$ for every pair of literals such that $j < k$ and σ_j is incompatible with σ_k.

In general, if each $\|A_j\| \le k$ and $k \ge 2$, the problem reduces directly to an instance of k`SAT` in which each literal occurs positively just once. The literals are (j, a) for $a \in A_j$, and the clauses are

$$\bigvee_{n \in A_j} (j,a)\,, \qquad \text{for } 1 \le j \le n\,;$$
$$\overline{(j,a)} \vee \overline{(k,b)}\,, \qquad \text{for } 1 \le j < k \le n \text{ and } a \text{ incompatible with } b.$$

Conversely, any instance of k`SAT` with m clauses reduces to the compatibility problem of finding representatives $(x_1, \ldots, x_m)$, with x_j a member of the jth clause and with two literals compatible iff they aren't negatives of each other.

The general compatibility problem with finite sets A_j can also be reduced to an independent set problem in a natural way. Consider the graph G with vertices (j, a) for $a \in A_j$, having edges

$$(j,a) — (j,b)\,, \qquad \text{if } a \ne b;$$
$$(j,a) — (k,b)\,, \qquad \text{if } j < k \text{ and } a \text{ is incompatible with } b.$$

Then G has an independent set of size n if and only if the compatibility problem has a solution.

Therefore we obtain simple solutions of the compatibility problem when there is a simple solution to the corresponding independent set problem. One such case occurs when compatibility is a symmetric relation and we have the following condition: If $i < j < k$ and the elements a_i, a_j, a_k are mutually compatible, then (1) every element of A_i is compatible with either a_j or a_k; (2) every element of A_j is compatible with either a_i or a_k; (3) every element of A_k is compatible with either a_i or $a-j$; and (4) every element not in $A_i \cup A_j \cup A_k$

is compatible with either x_i, x_j, or x_k. In such a case the graph G is claw-free, and we can use Minty's algorithm [7] to find a maximum independent set.

Grötschel, Lovász, and Schrijver [2, Chapter 9] have compiled a survey of cases where the independent set problem is known to have a simple solution.

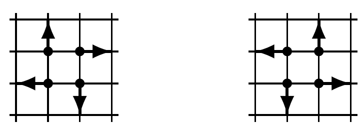
Another hard case. A very special case of the general mapmaker's problem, alluded to in the introduction, turns out to be NP-complete.

Consider a set of integer points $p_1, \ldots, p_n$ on the plane. We wish to find integer points $x_1, \ldots, x_n$ with the following properties for all $j \neq k$:

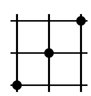
$$|x_j - p_j| = 1 \,; \qquad |x_j - p_k| > 1 \,; \qquad |x_j - x_k| \geq 2 \,.$$

(Motivation: Each x_j is the center of a 2×2 square in which a "label" for point p_j can be placed. The label at x_j should be closer to p_j than to any other point; distinct labels should not overlap.) We will call this the MFL problem, for "METAFONT labeling," because it arises in connection with the task of attaching labels to points in diagrams drawn by METAFONT [5, page 328].

Solutions to the MFL problem can conveniently be represented by showing each point p_j as a heavy dot and drawing an arrow from p_j to x_j for each j; at most four possibilities exist from each of the given points. For example, it is easy to see that a cluster of four adjacent points can be labeled in only two ways:

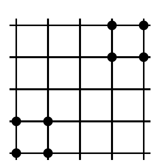


There is no way to attach a label to the middle point in a configuration like



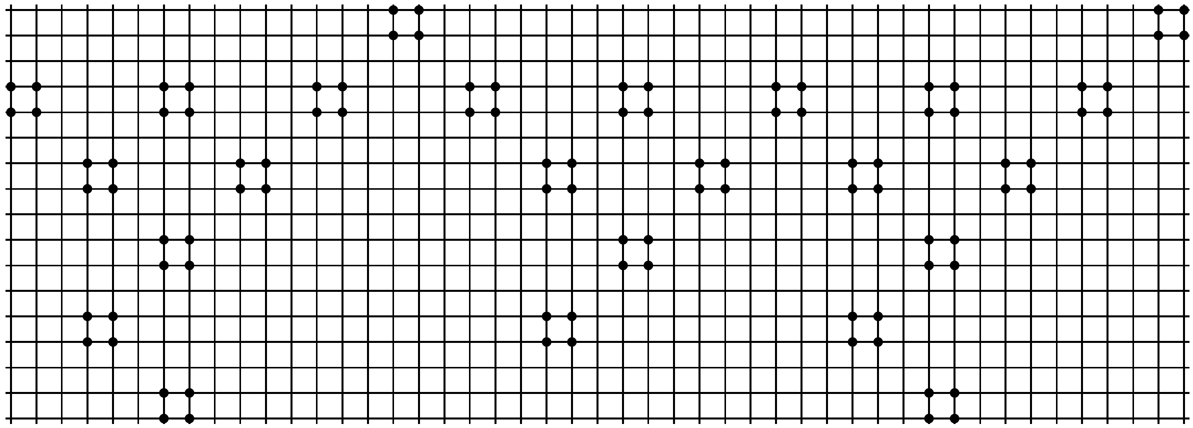
because each of the four positions adjacent to that point is too close to one of the other given points. The MFL problem provides an amusing pastime for people who are sitting in a boring meeting and who happen to have a tablet of graph paper to doodle on.

The general MFL problem is clearly in NP. In order to show that it is NP-complete, we observe first that there are only two solutions to the problem



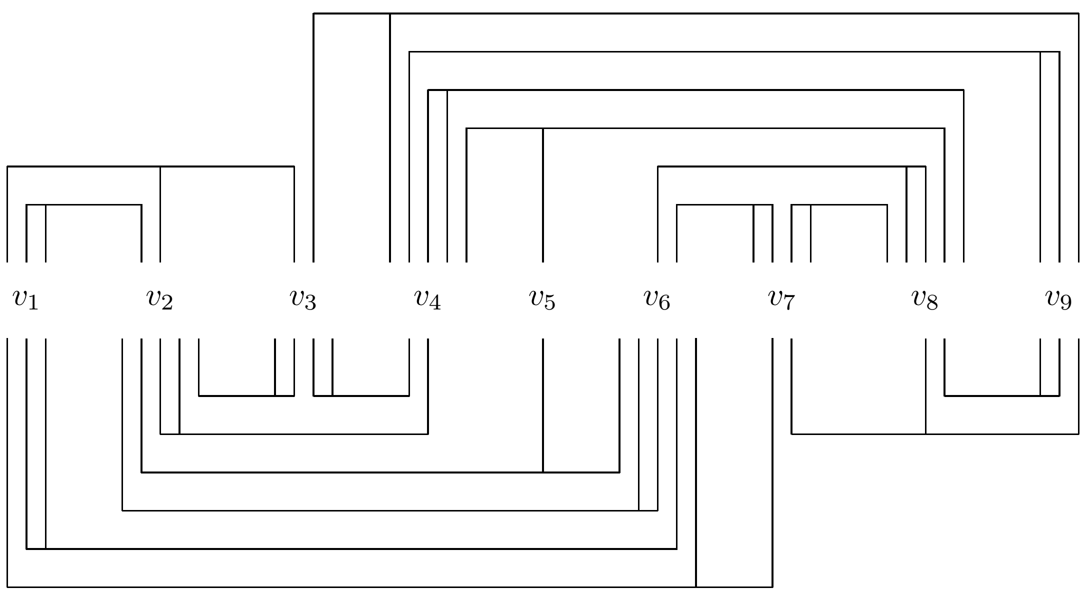
namely the two solutions for four-point clusters given earlier, using the same orientation in each cluster. Thus we can construct large chainlike tree networks of four-point clusters,

for example,



in which there are only two solutions, "positive" and "negative." This construction provides a way to represent the values of boolean variables in a satisfiability problem.

We can now use Lichtenstein's theorem that planar `3SAT` is NP-complete [6]. An instance of planar `3SAT` is a set of variables $v_1, \ldots, v_n$ arranged in a straight line, together with a set of three-legged clauses above and below them, where the clauses are properly nested so that none of the legs between clauses and variables cross each other. We can always put the clauses into a rectilinear configuration such as

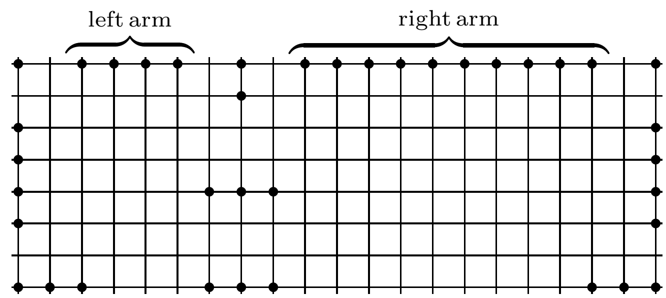


which corresponds to Lichtenstein's "crossover box" [6, Figure 5].

We construct an instance of `MFL` from a given instance of planar `3SAT` by representing the vertical legs for each variable as chains of four-point clusters; this guarantees that each variable will have one of two values, corresponding to the common orientation of all its

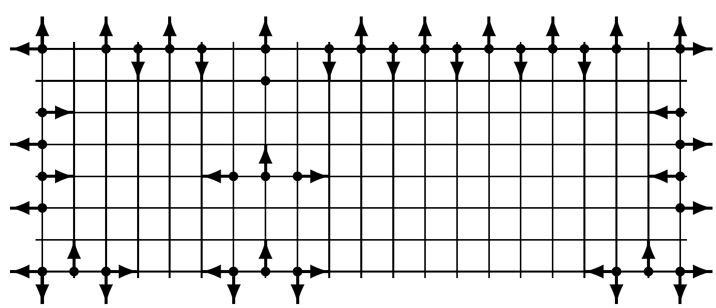
clusters. We can easily stretch out the diagram so that there is no interference between the variables except at places where three legs of a clause come together in a horizontal segment.

It remains to specify the representation of the clauses. By symmetry we need only describe the representation that appears above the variables. Each horizontal section of a comb-like clause in the upper portion will be represented by a configuration of the form



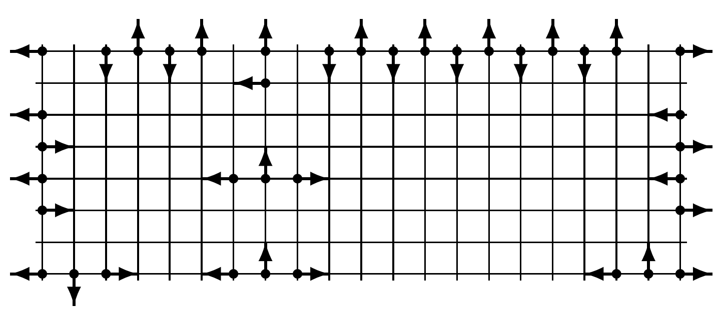
with $6l+4$ dots in the left arm and $6m+4$ dots in the right arm, for some l and m. (The three triples at the bottom will connect to clusters that represent variables, as explained below. Those clusters will occur at positions that are congruent mod 6; the arms of a comb stretch out so that they reach the variables appropriate to the clause.)

In each group of three dots at the bottom of this construction, the arrow for the middle dot must go either up or down. All three middle arrows cannot go up, because that forces

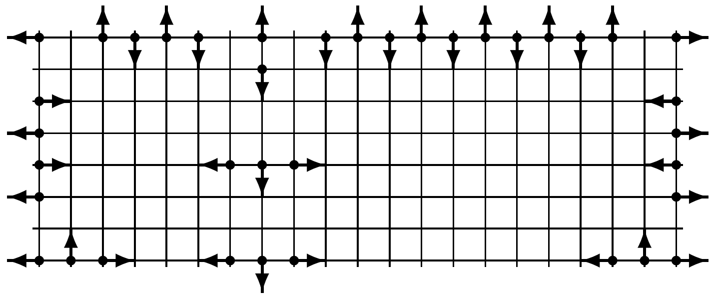


and there is no way to attach an arrow to the middle dot in the second row.

However, there are solutions in which any one of the middle arrows goes down. For example, we can choose

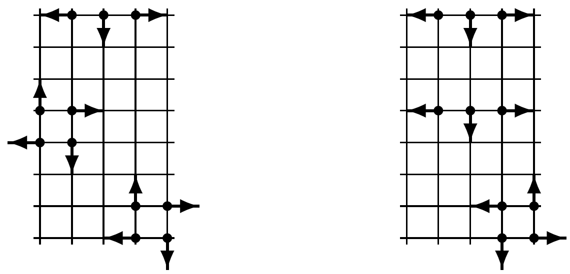


or



and there is a third solution that is essentially a mirror image of the first.

We can place four-point clusters below a row of three dots in such a way that a downward arrow on the top middle dot forces an orientation on the clusters, but an upward arrow on the top middle dot forces nothing:



By choosing one of these junction configurations for each variable in the clause, depending on whether the variable is negated or not, we obtain an instance of `MFL` that has a solution if and only if the given planar clauses are satisfiable.

Backtracking. We have now proved that `MFL` is NP-hard. However, in practice a solution or proof of nonexistence can often be found quickly by backtracking, using the idea of "preclusion" introduced by Golomb and Baumert [1]. When a trial value x_j is selected from A_j, it precludes all selections of other x_k that are incompatible with it; precluded values can be (temporarily) removed from A_k. The problem of compatible representatives is precisely the abstract general setting that supports this notion of preclusion.

Golomb and Baumert suggest choosing x_j at each stage from a currently smallest set A_j whose representative has not yet been chosen. If we are simply looking for a solution, instead of enumerating all solutions, it would also be worthwhile to select elements that preclude as few others as possible.

For example, if an element of A_j doesn't preclude any others, we can set x_j equal to that element without loss of generality. If $x \in A_j$ precludes only one element $y \in A_k$ and no others, and if we find no solution when $x_j = x$, then we can set $x_k = y$ without loss of generality.

Further work. A recent paper by Simon [8] considers the assignment of channels to transmitters in a radio communication system. This is another case of a compatibility problem,

rather like the mapmaker's problem because nearby transmitters must not broadcast on the same channel. Simon presents a polynomial-time approximation scheme that is guaranteed to find at least a fixed fraction of the optimum number of compatible channels. This suggests that many useful approximation schemes for other instances of the general compatibility problem might remain to be found.

References

[1] Solomon Golomb and Leonard D. Baumert, *Backtrack programming*, Journal of the ACM, 12 (1965), pp. 516–524.

[2] Martin Grötschel, László Lovász, and Alexander Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, New York, 1987.

[3] Marshall Hall, Jr, *Combinatorial Theory*, Blaisdell, 1967. Second edition, Wiley-Interscience, 1986.

[4] P. Hall, *On representatives of subsets*, Journal of the London Mathematical Society, 10 (1935), pp. 26–30.

[5] Donald E. Knuth, *The METAFONTbook*, Addison-Wesley, Reading, Mass., 1986.

[6] David Lichtenstein, *Planar formulæ and their uses*, SIAM Journal on Computing, 11 (1982), pp. 329–343.

[7] G. Minty, *On maximal independent sets of vertices in claw-free graphs*, Journal of Combinatorial Theory (B), 28 (1980), pp. 284–304.

[8] Hans Ulrich Simon, *Approximation algorithms for channel assignment in cellular radio networks*, in Fundamentals of Computation Theory, Proceedings of FCT'89, edited by J. Csirik, J. Demetrovics, and F. Gécseg, Lecture Notes in Computer Science 380 (1989), pp. 405–415.